Security vulnerabilities in DNS and DNSSEC

Suranjith Ariyapperuma and Chris J. Mitchell
Information Security Group
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
Email: suranjith.ariyapperuma@anglia.ac.uk, c.mitchell@rhul.ac.uk

Abstract

We present an analysis of security vulnerabilities in the Domain Name System (DNS) and the DNS Security Extensions (DNSSEC). DNS data that is provided by name servers lacks support for data origin authentication and data integrity. This makes DNS vulnerable to man in the middle (MITM) attacks, as well as a range of other attacks. To make DNS more robust, DNSSEC was proposed by the Internet Engineering Task Force (IETF). DNSSEC provides data origin authentication and integrity by using digital signatures. Although DNSSEC provides security for DNS data, it suffers from serious security and operational flaws. We discuss the DNS and DNSSEC architectures, and consider the associated security vulnerabilities.

Keywords: DNS, DNSSEC, Cryptography, Security.

1 Introduction

DNS [22, 24] is the standard mechanism for name to IP address resolution. For practical security and availability reasons it is important that DNS is able to tolerate failures and attacks. This is evident from recent phishing attacks [16] that have used DNS cache poisoning to steal sensitive financial data [21].

To solve the known security problems with DNS, a set of security extensions (DNSSEC) [5] have been proposed. DNSSEC provides data integrity and origin authentication using pre-generated digital signatures for each data item stored in the DNS database. However, as we show in this paper, DNSSEC introduces new security issues such as chain of trust problems, timing and synchronisation attacks, Denial of Service amplification, increased computational load, and a range of key management issues.

2 DNS Infrastructure

DNS translates domain names to IP addresses, and vice versa. DNS is implemented as a globally distributed database supporting a hierarchical structure. A client entity known as a Resolver acts on behalf of a client by submitting queries to, and receiving responses from, the DNS server. The responses contain Resource Records (RRs) containing the desired name/address resolution information. The availability and performance of DNS is enhanced through a replication and caching mechanism. A detailed description of the operation of DNS can be found in many books (see, for example, [23]).

The DNS name space is organised hierarchically. A domain is a subtree of the name space. The top-level domains (TLDs) are those immediately below the root. A domain is broken into smaller units called zones.

2.1 Name Servers and Resolvers

Name servers generally maintain complete name/address information about a particular zone in a file known as the zone file. Every zone needs to provide a primary and a secondary name server to enhance resiliency, Redundancy and Load Balancing. Primary master servers maintain the zone data in a locally stored zone file. All changes to zone data must take place at the database of the primary server. A secondary server periodically queries the master server for a copy of the database updates, and uses the returned information to update its own database. This is called zone transfer. The primary protocol used for DNS communications is UDP. However, TCP is used to provide zone transfers. Name servers listen on UDP and TCP port 53 for DNS queries.

In the absence of any other information, DNS query resolution must start with a query to the root name

Second International Conference on Availability, Reliability and Security (ARES'07)
'5-2/07 $20.00 © 2007 IEEE

servers, which means that the root servers are very busy. Caching is used extensively by all name servers to reduce the load on the root servers.

2.2 DNS Query Resolution

A name server can operate in two modes, recursive or iterative. A recursive query is sent with the RD (Recursion Desired) flag set to on in the DNS query header. In recursive mode the name server searches through the DNS hierarchy in response to queries and returns either an error or the answer, but never referrals to other name servers.

For iterative queries, the queried name server operating in iterative mode consults its own database for the requested data. If it cannot find the answer, it typically gives the IP address of the closest name server that might know the result. The client repeats the request, this time sending it to the server it just learned about. By default, queries to root name servers are iterative.

2.3 DNS Protocol Format

Figure 1 shows the format of DNS query and reply messages [18, 19]. We discuss the relevant fields below.

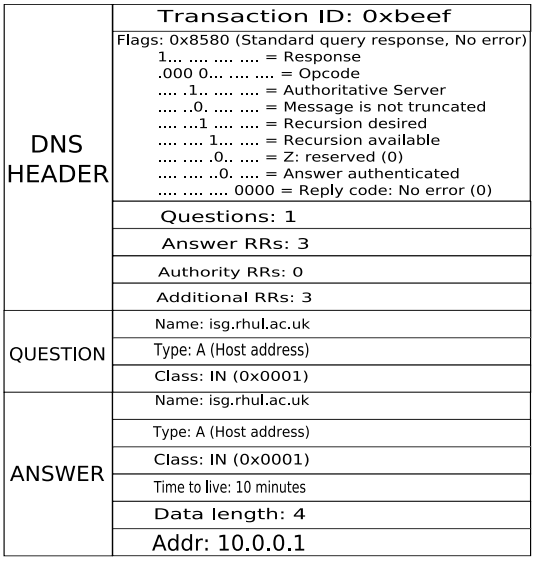

Figure 1. DNS Protocol Format

2.4 Resource Records

Resource Records (RRs), as stored in name server zone files, have the format shown in Figure 2

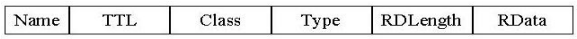

Figure 2. Resource Record Format

The domain name is a pointer to the RR. The Time To Live (TTL) is the length of time that a cached RR may be considered to be valid. Class is the type of the network; RRs typically belong to the IN (Internet) class. The most commonly used values for the Type field [19] are listed in Table 1.

Table 1. Resource Record Format

Type	Description
SOA	Start of Authority
NS	Name Server Record
A	Address Record (IP)
PTR	Pointer Record
MX	Mail Exchanger Record
CNAME	Canonical Name (Alias)

RDLength specifies the length of the RDATA field. Finally, RDATA describes the resource using the format specified by the type and class. An RRSet is a set of zero or more RRs [30], all of which have the same DNS name, class, and type.

3 DNS Vulnerabilities

We now briefly review some of the most important attacks on DNS. Most of these problems have been previously documented (see, for example, [6]).

3.1 Man in the middle (MITM) attacks

The recipient of data from a DNS name server has no way of authenticating its origin or verifying its integrity. This is because DNS does not specify a mechanism for servers to provide authentication details for the data they push down to clients. A resolver has no way to verify the authenticity and integrity of the data sent by name servers.

The Resolver can only authenticate the origin of a DNS reply data packet using the source IP address of the DNS server, destination and source port numbers and DNS transaction ID. An attacker can easily craft a DNS server response packet to match these parameters. The client has no choice but to trust as reliable the data provided by an attacker. An attacker can resolve legitimate queries, responding with false information.

3.1.1 Packet Sniffing

DNS sends an entire query or response in a single unsigned, unencrypted UDP packet, which makes it easy to tamper with. By capturing DNS query packets, a wrong answer can be generated fast enough to reach the resolver before the correct answer from the name server. Compromising a router on a transit network allows an attacker to capture the DNS Reply packet from the name server and modify it. As no source authentication or data integrity checks are supported, this will not be detected by the resolver.

3.1.2 Transaction ID Guessing

An attacker can respond with false answers to a predicted query, without having to be on the LAN to intercept packets. These answers will be cached either by the resolver or by the caching name server. The DNS Transaction ID field is only a 16-bit field, and the server UDP port associated with DNS is 53. On the client there are only 2^{32} possible combinations of ID (2^{16}) and client UDP ports (2^{16}) for a given client and server. In practice the client UDP port and the Transaction ID can be predicted from previous queries. It is common for the client port to be a known fixed value due to firewall restrictions, or the port number will increase incrementally due to resolver library behaviour. The DNS transaction ID generated by a client usually increases incrementally. This reduces the search space to a range smaller than 2^{16} [6].

By itself, ID guessing is not enough to allow an attacker to inject bogus data. This has to be combined with knowledge or guesses about Queries (QNAME) and Query type (QTYPE) for which a resolver might be querying. This can, for example, be achieved by cache snooping [13].

3.2 Caching Problems

Through the use of caches, the DNS sacrifices consistency in favour of reduced access time. DNS caching raises concerns about cache inconsistency and staleness of data. Stale information may include security critical information, e.g. a compromised key. The current DNS protocol does not support any means to propagate data updates or invalidations to DNS servers or caches in a fast and secure way.

3.2.1 Cache Poisoning using Name Chaining

This attack introduces false information into DNS caches. This is achieved by means of DNS RRs whose RDATA portion includes a DNS name which can be used as a hook to let an attacker feed bad data into a victim's cache. The most affected types of RRs are CNAME, NS, and DNAME RRs.

False data, associated with these names, can be injected into the victim's cache via the Additional section of the response. An Attacker can introduce arbitrary DNS names of the attacker's choosing, and provide further information that is claimed to be associated with those names [6].

3.2.2 Cache Poisoning using Transaction ID Prediction

In this attack, a large number of resolution requests are sent to the victim server (ns1.victim.com, say) with spoofed source IP addresses to resolve a name, say www.mybank.com. Each request will be assigned a unique transaction ID and processed independently. Since ns1.victim.com is trying to resolve each of these requests, the server will be awaiting a large number of replies from ns1.mybank.com.

The attacker uses this wait stage to bombard ns1.victim.com with spoofed replies from ns1.mybank.com, stating that www.mybank.com points to an IP address which is under the attacker's control. Each spoofed reply has a different transaction ID, a source port and the spoofed DNS server IP address (for ns1.mybank.com). The attacker hopes to guess the correct transaction ID and source port used by the querying name server. Once the attack is successful, false information will be stored in ns1.victim.com's cache [26].

3.3 DDoS attacks

DDoS attacks can have a significant impact on the global DNS database and its users. They are usually directed at root servers. This was evident with the recent DDoS attack in June 2004 [28], which was a repeat of a similar attack in October 2002 [20]. These attacks

caused a loss of availability of name resolution services to the Internet community.

3.4 Other significant DNS attacks

3.4.1 Information Leakage

A successful zone transfer by an attacker would serve as a reconnaissance attack, potentially revealing sensitive information about internal network configuration, e.g. the IP addresses of Internal firewall interfaces. DNS names could, for example, represent project names that may be of interest to an attacker, or could reveal the identity of the operating system running on the machine.

3.4.2 DNS Dynamic Update Vulnerabilities

Protocols such as Dynamic Host Configuration Protocol (DHCP) make use of the DNS Dynamic Update protocol to add and delete RRs on demand. These updates take place on the primary server of the zone [30]. The authentication for such updates is based solely on source IP address, and is vulnerable to threats such as IP spoofing. These attacks can range from denial of service, including deletion of records, to redirection [1].

3.5 BIND Security Considerations

DNS servers across the Internet using the BIND implementation of DNS software are not always up to date with security patches and software updates. As a result, at any one time a significant fraction of the Internet's DNS servers are vulnerable to compromise [27]. The majority of these vulnerabilities are a result of poor exception handling and boundary checking. Exploitation will potentially allow attackers to execute arbitrary code or write data to arbitrary locations in memory [11].

3.6 Usage vulnerabilities

The use of DNS triggers conditions similar to DDoS attacks. The DNS query rate at the root servers has risen from 1 query per second to roughly 100000 queries per second in 2004 [17, 25]. Measurements at root servers show an astounding number of bogus queries: from 60-85 % of observed queries were repeated from the same host within the measurement interval. Over 14 % of a root server's query load is due to queries that violate the DNS specification [8].

Root servers receive a large number of queries because the resolvers never receive the replies, because of either packet filters or routing issues [32]. Improper usage or improper coding of the client resolvers can cause a DDoS effect on DNS root servers.

4 Domain Name System Security Extensions (DNSSEC)

DNSSEC adds security to the DNS protocol by providing origin authentication, data integrity and authenticated denial of existence to DNS data provided by a name server. All answers from DNSSEC servers are digitally signed. By checking the signature, a DNSSEC resolver is able to check if the information originated from a legitimate server and that data is identical to the data on the authoritative DNS server. If the data is not present on the server an authenticated denial is produced.

To maintain backward compatibility with DNS, DNSSEC requires only minor changes to the DNS protocol. DNSSEC adds four record types to DNS, namely Resource Record Signature (RRSIG), DNS Public Key (DNSKEY), Delegation Signer (DS), and Next Secure (NSEC). DNSSEC uses two of the previously unused flag bits in the DNS query and answer message header (AD and CD). The AD (Authentic Data) bit in a response indicates that all the data included in the answer and authority portion of the response has been authenticated by the server. The CD (checking disabled) bit indicates that unauthenticated data is acceptable to the resolver sending the query [5]. Because the UDP protocol has a packet size limit of 512 bits, DNSSEC requires the use of EDNS0 [29] extensions that override this limitation, so that larger key sizes can be accommodated [10].

DNSSEC adds the ability to detect MITM attacks on DNS through the addition of data origin authentication, transaction and request authentications, but DNSSEC does not prevent such attacks. To maintain data origin authenticity and integrity, both servers and resolvers must use the DNSSEC protocol.

4.1 Keys in DNSSEC

Each secured zone has a key pair, made up of a zone private key and the corresponding public key. The zone public key is stored as a resource record (type KEY) in the secured zone. The public key is used by DNS servers and Resolvers to verify the zone's digital signature.

All resource records in a secured zone are signed by the zone's private key. To make zone re-signing and key roll-overs easier to implement, it is possible to use one or more keys as Key Signing Keys (KSKs). A KSK

will only be used to sign the top level KEY RRs in a zone. Zone Signing Keys (ZSKs) are used to sign all the RRsets in a zone.

4.2 Signatures in DNSSEC

DNSSEC provides an unforgeable authentication of a RRset by associating it with a signature resource record that binds DNS data to a time interval and the signer's domain name.

A private key is used to sign a RRset. For increased speed a hash of the RRset is signed. This provides authenticated data origin. If data is modified during transport the signature is no longer valid (authenticated data integrity). In DNSSEC, only signatures are used, and nothing is encrypted.

Hashes are generated using MD5 or SHA-1. Signatures are created using MD5/RSA [4], DSA [3] or elliptic curve cryptographic algorithms. Signatures are stored as resource records (type RRSIG) and are used with the zone's public key to authenticate resource records.

4.3 NSEC Records

Each unique name in a secured zone is also assigned a corresponding NSEC resource record, which points to the next name present in the zone. The sequential chain of NSEC resource records for a zone defines what resource records actually exist in that zone. The NSEC resource records are also signed by the zone private key, preventing the zone from being compromised through unauthorised addition or deletion of zone resource records. NSEC resource records for a zone are automatically generated when the zone records are signed.

4.3.1 Time in DNSSEC

All times in DNS are relative. The Start Of Authority (SOA) resource record's refresh, retry and expiration timers are counters that are used to determine the time elapsed since a slave server synchronised with a master server. The Time to Live (TTL) value is used to determine how long a forwarder should cache data after it has been fetched from an authoritative server. DNSSEC introduces absolute time into DNS.

The signature validity period is the period that a signature is valid. It starts at the time specified in the signature inception field of the RRSIG and ends at the time specified in the expiration field. The signature publication period is the time after which a signature is replaced with a new signature by publishing the relevant RRSIG in the master zone file.

5 DNSSEC Vulnerabilities

DNSSEC does not guard against poor configuration or bad information in the authoritative name server, and does not protect against buffer overruns or DDoS attacks. A large increase in the computational load on the servers and resolvers, a hierarchical model of trust, the lack of management tools, and the need for a higher level of time synchronisation between the servers, remain some of the most significant obstacles to its deployment.

Cryptographic key management issues, such as initial key configuration and key rollover, key authentication and verification have yet to be resolved at an operational level in order to enable DNSSEC to be deployed on a global scale. Storage of the zone private key is also an issue, and DNSSEC cannot tolerate malicious server failures. Finally as SECREG [12] test bed experience suggests, DNSSEC is complex to implement.

5.1 The Chain of Trust

DNSSEC has a hierarchical trust model. To securely resolve a name in DNSSEC, a root public key must be available at the resolver. Public keys themselves are authenticated either by being statically configured into a DNS server or resolver, or by being signed by a parent zone's private key.

The chain of trust works by following secured delegation pointers called Delegation Signers (DSs). The DS record delegates trust from a parental key to a child's zone key. The DS record holds a SHA-1 hash of a child's zone public key, and this hash is signed with the zone private key of the parent. By checking the signature of the DS record, a resolver can validate the hash of the child's zone public key, and can in turn establish the validity of the child's public key.

A root public key injection attack at the client end, as described in [9], would compromise this chain of trust. Any compromise in any of the zones between the root and a particular target name will result in data being marked as bogus, which may cause entire sub-domains to become invisible to verifying clients.

Data published on an authoritative primary server will not be seen immediately by verifying clients; it may take some time for the data to be transferred to other secondary authoritative name-servers. During this period, clients may be fetching data from caching non-

authoritative servers. For the verifying clients it is important that data from secured zones can be used to build chains of trust regardless of whether the data came directly from an authoritative server or a caching nameserver [12].

5.2 Security Considerations for keys

The key size, algorithm and validity period should be chosen with care. The Key validity period [2] affects a number of issues with respect to the generation, lifetime, size and storage of private keys. When choosing key sizes, zone administrators need to take into account the key validity period and how much data will be signed during that period.

5.2.1 Key Rollovers

The longer a key is in use, the greater the probability that it will have been compromised through carelessness, accident, espionage, or cryptanalysis [2].

During the key rollover, data published about the previous versions of the zone still live in caches. Ignoring previously cached data will lead to loss of service for clients, especially when zone material signed with an old key is being validated by a resolver which does not have the old zone key cached. If the old key is no longer present in the current zone, this validation fails, marking the data bogus. Alternatively, an attempt could be made to validate data which is signed with a new key against an old key that lives in a local cache, also resulting in data being marked bogus.

Key rollover at the root is difficult to achieve as it affects the whole hierarchical database. Work needs to be done to adequately specify how the root key rolls over.

5.2.2 Zone Private Key Storage

DNSSEC recommends that the zone private key should be stored offline. This creates issues with providing support for dynamic updates, as the zone private key cannot be used to generate signatures (RRSIG) in real-time to authenticate dynamically updated data. This makes dynamically updated data in DNSSEC either unavailable or unprotected before the next zone signing using the zone private key takes place.

A security breach of the server that stores the zone private key gives an intruder full access to the zone. To ensure full security, the zone private key should never be present in the server memory.

5.2.3 DNSSEC Timing issues

DNSSEC creates a requirement for loose time synchronisation between the resolver and the host creating the DNSSEC signatures. A resolver must have the same concept of absolute time in order to determine whether the signature is valid or expired. An attacker that can change a resolver's opinion of the current absolute time can fool the resolver into using expired signatures. An attacker that can change the zone signer's opinion of the current absolute time can fool the zone signer into generating signatures whose validity period does not match what the signer intended.

Signatures (RRSIG) are not generated on a per query basis, but are instead occasionally generated with a high TTL, as signing is a resource expensive process. This opens a broad window for a freshness attack, where stale data whose corresponding signature has not expired is replayed to misdirect the client.

SIG RRs are valid from a pre-set signature inception time to a signature expiration time. This interval tends to be long. The original TTL value of the RR is also included and protected by the signature. The corresponding RR is valid until the signature expiration time or the TTL expiration time, whichever comes first. Even if an RR is updated, the actual update remains ineffective until the signature or the TTL expiration time. DNSSEC requires at least some time synchronisation between the primary and secondary name servers.

5.2.4 Wildcard Proof Mechanism

A wildcard (*), used in the name field of a RR, allows one record to stand in for a number of other records (of the same type, pointing to the same data, in the same zone). Wildcard names are patterns for synthesising RRs on the fly, according to the matching rules [18]. Wildcard MX RRs are used heavily.

The existence of the wildcard RR, and the nonexistence of any RRs which, if they existed, would make the wildcard RR irrelevant (according to the synthesis rules), must be proven. This makes the wildcard proof mechanism dependent upon an authenticated denial mechanism. DNSSEC processes wildcard proofs as described above [6].

5.2.5 Increased Computational Load

DNSSEC significantly increases the size of DNS response packets, which drastically increases the computational load on the DNS servers and also increases the query response time. The process of verifying signed

resource records alone is computationally intensive, particularly for larger key sizes. The DNSSEC standard allows up to 1024-bit keys. Adding digital signatures to a domain increases each record size by 5-7 times, which puts a burden on upstream name servers [14].

DNSSEC answer validation increases the resolver's work load, since DNSSEC-aware resolvers will need to perform signature validation and, in some cases, will also need to issue further queries. This increased workload will also increase the time it takes to get an answer back to the original DNS client.

5.2.6 Lack of Management Tools

DNSSEC is significantly more complex than DNS. There are only a few tools to help with the task of maintaining a signed domain. DNSSEC monitoring and log analysis tools are still being developed. Debugging manually is a time consuming and difficult task.

5.2.7 Lack of Consistency Control

DNSSEC uses a simple primary-secondary replication scheme to provide server availability and load distribution. Only one of the primary or the secondary name server takes part in a particular query. One compromised server is enough to block, steal, or redirect information, and issue fraudulent update requests. Data update takes place only at the primary server, and the secondary server is not aware of the update until after the next zone transfer. Thus, there may be inconsistencies between the states of the primary and the secondary server, and a client query may result in completely different answers depending on which server served the query. This causes consistency and reliability problems even without an intentional attack.

5.2.8 DNSSEC NSEC zone walking vulnerability

NSEC is used for authenticated proof of non-existence of DNS RRs. This introduces the ability for a hostile party to enumerate all the names in a zone by following the NSEC chain. NSEC RRs assert which names do not exist in a zone by linking from existing name to existing name along a canonical ordering of all the names within a zone. An attacker can query these NSEC RRs in sequence to obtain all the names in a zone. This allows an attacker to enumerate the zone. Techniques such as minimally covering NSEC records [31] and NSEC3 [15] have been proposed to solve this problem.

5.2.9 DNSSEC Problems encountered in .NL

SEGREG is a DNSSEC experiment conducted by NLNetLabs [12]. During the experiment one of the .nl secondaries (dnssec.nic-se.se) was not updated for more than two weeks due to a memory overflow. This caused severe problems. The signature lifetime for .nl is one week, and thus the signatures on nic-se.se all expired. Users of this server got bad (authoritative) data for .nl. With DNSSEC, a secondary that is out of date for longer than the signature lifetime causes the local removal of a domain.

6 Summary and Conclusions

DNS has major security issues that need to be addressed urgently. Threats such as Man in the middle attacks and cache poisoning arise because of the lack of authentication and integrity in the DNS transaction process. Inaccurate or nonexistent boundary checking and error handling conditions in BIND software lead to exploits such as buffer overflows. Usage threats are caused by a range of entities from misconfigured client resolvers to packet filters causing conditions similar to DDoS.

The Internet Engineering Task Force (IETF) has responded to the threats by developing DNSSEC, a secure DNS protocol, to address the data integrity and source spoofing issues. DNSSEC allows transaction level authentication and secure zone transfers protecting all data in the zone during the transfer. In DNSSEC, Name-based authentication attacks can be detected [7].

DNSSEC does not protect against buffer overruns or DDoS attacks, nor does it provide confidentiality. Secure delegation is complex to implement, and DNSSEC's error reporting capabilities are minimal. DNSSEC zone files are significantly larger than their DNS counterparts, including the load on servers, network and resolver.

The public/private keys used with DNSSEC can be compromised over time. Keys should be changed at intervals to reduce the risk of compromise. This can be implemented relatively easily at the lower levels of the DNSSEC hierarchy, as the public keys are not cached for very long. Root key rollover however is a problem, as authentication (chain of trust) is based on known root keys. Regular key rollover has a significant impact on the entire DNSSEC structure. Selection of timing parameters is critical in DNSSEC, involving TTL, signature inception time and signature expiration time.

Despite the vulnerabilities in DNSSEC, it provides integrity and authentication to DNS data. DNSSEC is

backward compatible with the existing DNS infrastructure. We suggest further research in areas such as advanced dynamic zone transfer protocols using link state type algorithms, unification of alternate roots with the ICANN root to form a meta root, resiliency of DNSSEC, active attacks against distributed directory services such as X.500, Light Weight Directory Access Protocol (LDAP) and Kerberos, DNSSEC on mobile devices and MANETs, and DDoS protection against DNSSEC, to improve the core functionality of DNSSEC.

7 Acknowledgements

The first author would like to thank Anglia Ruskin University for their generous support.

References

[1] D. E. 3rd. Secure domain name system dynamic update. RFC 2137, Internet Engineering Task Force, Apr. 1997.

[2] D. E. 3rd. DNS security operational considerations. RFC 2541, Internet Engineering Task Force, Mar. 1999.

[3] D. E. 3rd. DSA KEYs and SIGs in the domain name system (DNS). RFC 2536, Internet Engineering Task Force, Mar. 1999.

[4] D. E. 3rd. RSA/MD5 KEYs and SIGs in the domain name system (DNS). RFC 2537, Internet Engineering Task Force, Mar. 1999.

[5] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS security introduction and requirements. RFC 4033, Internet Engineering Task Force, Mar. 2005.

[6] D. Atkins and R. Austein. Threat analysis of the domain name system (DNS). RFC 3833, Internet Engineering Task Force, Aug. 2004.

[7] S. M. Bellovin, J. Schiller, and C. Kaufman. Security mechanisms for the Internet. RFC 3631, Internet Engineering Task Force, Dec. 2003.

[8] N. Brownlee, K. Claffy, and E. Nemeth. DNS measurements at a root server. Technical report, Cooperative Association for Internet Data Analysis — CAIDA, San Diego Supercomputer Center, University of California, San Diego, 2001.

[9] D. W. Chadwick and G. Zhao, editors. *Public Key Infrastructure, Second European PKI Workshop: Research and Applications, EuroPKI 2005, Canterbury, UK, June 30 - July 1, 2005, Revised Selected Papers*, volume 3545 of *Lecture Notes in Computer Science*. Springer, 2005.

[10] B. Cohen. DNSSEC: Security for Essential Network Services. *Enterprise Networking Planet*, 2003.

[11] I. A. Finlay. CERT advisory CA-2002-15 denial-of-service vulnerability in ISC BIND 9. Internet, 2002.

[12] R. Gieben. DNSSEC in NL Final Report. Technical report, NLnet Labs, 2004.

[13] L. Grangeia. Dns cache snooping. Technical report, Securi Team — Beyond Security, February 2004.

[14] O. M. Kolkman. Measuring the resource requirements of dnssec. Technical report, RIPE NCC / NLnet Labs, October 2005.

[15] B. Laurie, G. Sisson, R. Arends, and D. Blacka. Dnssec hashed authenticated denial of existence draft-ietf-dnsext-nsec3-06. Technical report, Internet Engineering Task Force, June 2006.

[16] J. Leyden. Phishing morphs into pharming. Technical report, www.theregister.co.uk.

[17] P. Mockapetris and K. J. Dunlap. Development of the domain name system. In *SIGCOMM '88: Symposium proceedings on Communications architectures and protocols*, pages 123–133, New York, NY, USA, 1988. ACM Press.

[18] P. V. Mockapetris. Domain names — concepts and facilities. RFC 1034, Internet Engineering Task Force, Nov. 1987.

[19] P. V. Mockapetris. Domain names — implementation and specification. RFC 1035, Internet Engineering Task Force, Nov. 1987.

[20] R. Naraine. Massive DDoS attack hit DNS root servers. Technical report, Internet News — www.internetnews.com, 2002.

[21] G. Ollmann. The phishing guide. Technical report, Next Generation Security Software (NGS), 2005.

[22] C. Partridge and G. Trewitt. HEMS variable definitions. RFC 1024, Internet Engineering Task Force, Oct. 1987.

[23] C. L. Paul Albitz. *DNS and BIND*. O'Reilly, 2001.

[24] J. B. Postel. TCP and IP bake off. RFC 1025, Internet Engineering Task Force, Sept. 1987.

[25] Reseaux IP Europeens (RIPE) NCC. K-root statistics, k.root-servers.net. Technical report, Reseaux IP Europeens, 2004.

[26] Sainstitute. Attacking the DNS Protocol Security Paper v2. Technical report, Security Associates Institute, 2003.

[27] M. Schiffman. Bound by tradition: A sampling of the security posture of the internet's DNS servers. Internet, 2003.

[28] J. Udell. Needed: Rapid internet response. Technical report, Infoworld — www.infoworld.com, 2004.

[29] P. Vixie. Extension mechanisms for DNS (EDNS0). RFC 2671, Internet Engineering Task Force, Aug. 1999.

[30] P. Vixie, S. Thomson, Y. Rekhter, and J. Bound. Dynamic updates in the domain name system (DNS UPDATE). RFC 2136, Internet Engineering Task Force, Apr. 1997.

[31] S. Weiler and J. Ihren. Minimally covering NSEC records and DNSSEC on-line signing. RFC 4070, Internet Engineering Task Force, Apr. 2006.

[32] D. Wessels and M. Fomenkov. Wow, that's a lot of packets. Technical report, Cooperative Association for Internet Data Analysis — CAIDA, San Diego Supercomputer Center, University of California, San Diego, 2003.